

КАЗАХСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ
ИМЕНИ АЛЬ-ФАРАБИ

ЛЕКЦИЯ № 6

Современные аппаратные платформы искусственного интеллекта (AI Hardware). Технические характеристики решений NVIDIA и конкурирующих платформ

Бельгибаев Бауржан Абдрахимович
Д.т.н., доцент · Кафедра «Искусственный интеллект и big data»
«Технологии в интеллектуальных системах управления в IoT» · 4 курс, бакалавриат

1 Цель лекции

Сформировать системное представление об аппаратной архитектуре нейронных мини-компьютеров и интеллектуальных контроллеров, научить обоснованно выбирать вычислительные платформы для обработки сенсорных данных и принятия управляющих решений в IoT-системах с учётом ограничений реального времени, энергопотребления, стоимости и надёжности.

2 План лекции

1. Место AI Hardware в интеллектуальных системах управления
2. Архитектуры CPU, GPU, NPU, TPU и FPGA
3. Платформы NVIDIA Jetson и программная экосистема
4. Конкурирующие платформы Edge AI и критерии сравнения
5. Математические показатели производительности и задержек
6. Развёртывание нейронной модели на встраиваемой платформе
7. Практический пример: выбор платформы для мобильного IoT-робота
8. Методика экспериментальной проверки и выводы

1 Место AI Hardware в интеллектуальных системах управления

Интеллектуальная система управления (ИСУ) объединяет измерение состояния объекта, обработку информации, принятие решения и формирование управляющего воздействия. В традиционной автоматизации логика регулятора задаётся аналитической моделью или правилами. В системах с искусственным интеллектом часть обработки выполняется обученной моделью: классификатором, детектором объектов, оценщиком состояния либо прогнозирующим алгоритмом. Аппаратное обеспечение AI Hardware определяет, можно ли выполнить такой алгоритм непосредственно на объекте, не отправляя каждый кадр или измерение в облако.

Типовой контур Edge AI: датчики и камера → интерфейсы ввода → предобработка → нейронный ускоритель → постобработка → модуль принятия решений → контроллер исполнительных механизмов → объект управления. Отдельный контур обратной связи измеряет результат действия. Облачная инфраструктура может использоваться для обучения, архивирования и диспетчеризации, но критические функции безопасности желательно сохранять локально.

1 Место AI Hardware в интеллектуальных системах управления (продолжение)

Различают облачные, пограничные (edge) и конечные (tiny/endpoint) вычисления. Облачная архитектура предоставляет значительные ресурсы, но зависит от связи. Edge-платформа размещается рядом с объектом и снижает сетевую задержку. TinyML позволяет выполнять небольшие модели на микроконтроллерах с ограниченной памятью и мощностью. Выбор зависит от частоты измерений, размера модели, допустимой задержки, условий окружающей среды и требований к автономности.

1.1. Ограничения промышленного применения

Для управления недостаточно высокой скорости распознавания в среднем: необходимо оценивать наихудшую задержку, периодичность выполнения задач, устойчивость к потере кадров и безопасный переход в резервный режим. Нейросетевой вывод не заменяет аппаратные цепи аварийной остановки, защитные блокировки и детерминированные функции ПЛК. Для промышленного объекта требуется разделение критического контура управления и некритической аналитики.

2 Архитектуры вычислительных устройств

2.1. Центральный процессор (CPU)

CPU предназначен для последовательного и смешанного исполнения задач, работы операционной системы, сетевого стека, логики управления и подготовки данных. Его сильная сторона — гибкость и относительно предсказуемая обработка небольших алгоритмов. Однако большие матричные операции нейронных сетей часто выполняются менее энергоэффективно, чем на специализированных ускорителях.

2.2. Графический процессор (GPU)

GPU содержит множество вычислительных блоков для параллельной обработки тензоров. В нейронных сетях основная нагрузка возникает при свёртках и матричных умножениях. GPU полезен для детекции объектов, сегментации, визуальной одометрии и обработки нескольких видеопотоков. При проектировании следует учитывать объём доступной памяти, пропускную способность памяти, режим энергопотребления и поддержку программных библиотек.

Архитектуры вычислительных устройств (продолжение)

2.3. Нейронные ускорители (NPU, TPU)

NPU и TPU — специализированные вычислительные блоки для операций машинного обучения. Они обычно демонстрируют высокую эффективность при поддерживаемых типах слоёв и точности представления чисел. Ограничения возникают при неподдерживаемых операторах, фиксированных размерах тензоров, преобразовании формата модели и необходимости выполнять часть графа на CPU.

2.4. FPGA и микроконтроллеры

FPGA допускает создание специализированных конвейеров обработки сигналов с низкой задержкой, но требует сложной разработки и верификации. Микроконтроллеры с DSP-инструкциями и встроенными нейронными блоками подходят для классификации сигналов, диагностики вибраций и простых задач распознавания. Важно различать микроконтроллер, одноплатный компьютер и внешний USB/PCIe-ускоритель: это разные классы устройств.

2 Архитектуры вычислительных устройств (продолжение)

Тип	Преимущество	Ограничение	Типовая роль
CPU	Универсальность	Низкая эффективность больших нейросетей	Логика и предобработка
GPU	Массовый параллелизм	Тепло и энергопотребление	Компьютерное зрение
NPU/TPU	Высокая энергоэффективность	Зависимость от операторов и SDK	Нейросетевой вывод
FPGA	Настраиваемый аппаратный конвейер	Сложность проектирования	Сигналы и низкие задержки
MCU/TinyML	Малая мощность и стоимость	Ограниченная память	Сенсорные узлы

4 Спектр вычислительных архитектур

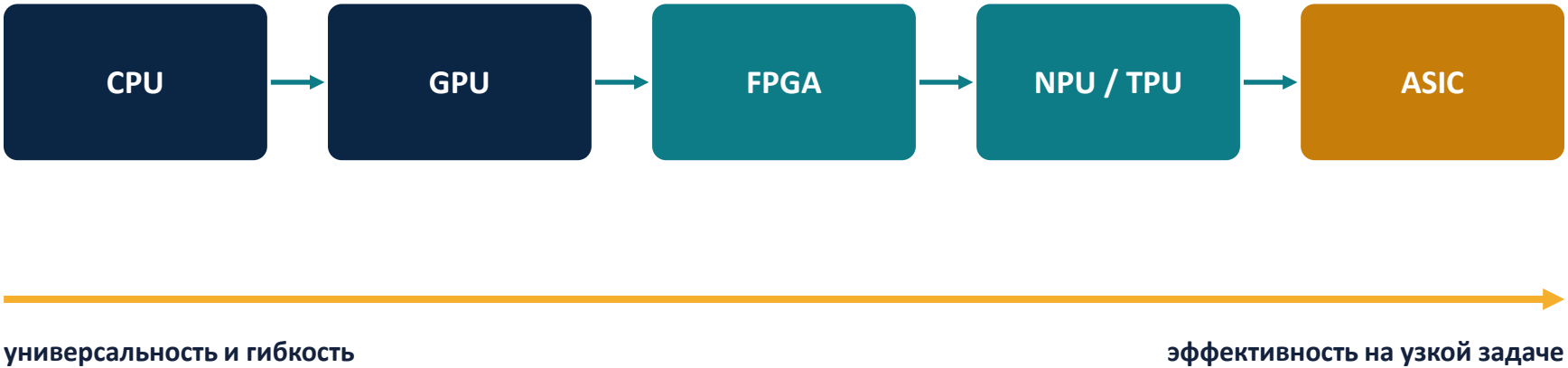

Рис. 10. Чем специализированнее архитектура, тем выше эффективность и ниже универсальность.

3 Платформы NVIDIA Jetson и программная экосистема

Семейство NVIDIA Jetson объединяет ARM-процессоры, графические ускорители и интерфейсы периферии на одной вычислительной платформе. Оно ориентировано на робототехнику, видеоаналитику и автономные устройства. К известным поколениям относятся Jetson Nano, Xavier NX, AGX Xavier, Orin Nano, Orin NX и AGX Orin; более новые решения следует оценивать по актуальным официальным спецификациям. Название семейства само по себе не определяет производительность: она зависит от конкретной модификации, профиля мощности, программного стека и модели.

Программная экосистема включает JetPack, CUDA, TensorRT, инструменты видеодекодирования и интерфейсы интеграции с робототехническим ПО. TensorRT позволяет оптимизировать граф нейронной сети для выполнения на поддерживаемом GPU, включая объединение операций и использование пониженной точности, если это допускает модель и требования к качеству.

При оценке Jetson нужно фиксировать не только пиковую вычислительную производительность, но и рабочий профиль мощности, используемую точность INT8/FP16/FP32, температуру, размер входного изображения, версию программной среды и реальные задержки. Показатели TOPS разных производителей нельзя непосредственно сопоставлять без одинаковых условий и тестовых моделей.

Платформы NVIDIA Jetson и программная экосистема (продолжение)

3.1. Последовательность развёртывания

- Подготовить совместимый образ ОС и версии JetPack для конкретной платы.
- Проверить доступность камеры, сетевых интерфейсов и системы охлаждения.
- Экспортировать обученную модель в совместимый формат, например ONNX.
- Проверить поддерживаемые операторы и построить оптимизированный движок вывода.
- Измерить точность модели после преобразования и квантования.
- Выполнить профилирование задержки, загрузки памяти, мощности и температуры.
- Интегрировать результат распознавания с независимым контроллером исполнительных механизмов.

4 Конкурирующие платформы Edge AI

Google Coral использует Edge TPU для выполнения совместимых квантованных моделей TensorFlow Lite. Платформа удобна для компактных задач вывода, но требует проверки совместимости операций и правильной компиляции модели. Raspberry Pi без внешнего ускорителя может выполнять небольшие модели на CPU, а с совместимым AI-ускорителем — переносить часть вычислений на специализированный блок. Устройства на базе Qualcomm, Hailo, Intel и других производителей предоставляют собственные ускорители и программные среды. Выбор нельзя сводить к рекламируемым TOPS: важны доступность драйверов, совместимость модели и жизненный цикл изделия.

4 Конкурирующие платформы Edge AI (продолжение)

Платформа / класс	Типичный сценарий	Что проверить
NVIDIA Jetson	Мобильная робототехника, несколько камер	Память, JetPack, TensorRT, режим мощности
Google Coral Edge TPU	Компактный INT8-вывод	Совместимость TFLite и операторов
Raspberry Pi + AI accelerator	Прототипы IoT и видеоаналитика	Шина подключения, SDK, питание
Intel AI/встраиваемые платформы	Машинное зрение и промышленная аналитика	Поддержка OpenVINO и конкретного чипа
MCU с TinyML	Сенсорная классификация	RAM, Flash, время инференса

5 Математические показатели производительности

5.1. Задержка обработки

Полная задержка от измерения до управляющего воздействия складывается из времени захвата, предобработки, инференса, постобработки, передачи команды и реакции исполнительного устройства:

$$T_total = T_capture + T_pre + T_infer + T_post + T_comm + T_act.$$

Для периодического контура с периодом T_s необходимо, чтобы наихудшее время обработки не превышало выделенный временной бюджет. Среднее значение задержки само по себе не гарантирует выполнение этого условия.

5.2. Частота обработки и пропускная способность

Для последовательного исполнения одного кадра приближённая частота FPS ≈ 1 / T_frame, если время выражено в секундах. В конвейере с параллельными этапами производительность ограничивается самым медленным этапом и накладными расходами; поэтому измеренные FPS и задержка отдельного кадра не всегда являются обратными величинами.

5 Математические показатели производительности (продолжение)

5.3. Энергоэффективность

Энергия одного вывода: $E_{infer} = P_{avg} \cdot T_{infer}$, где P_{avg} — средняя мощность за интервал вычисления. Энергоэффективность можно оценить количеством корректно обработанных кадров на джоуль, однако необходимо учитывать энергопотребление всей системы, включая камеру, память и охлаждение.

5.4. Память и качество модели

Объём памяти для параметров модели приблизительно равен $M_{weights} = N_{params} \cdot b / 8$, где N_{params} — число параметров, b — число бит на параметр. Дополнительно требуются память для активаций, буферов ввода и программной среды. Квантование снижает память и иногда ускоряет вывод, но способно ухудшить точность; это проверяется на отдельной тестовой выборке.

Для задач классификации используют accuracy, precision, recall и F1. Для детекции объектов применяют IoU и mAP. В задачах управления дополнительно оценивают частоту ложных тревог, вероятность пропуска опасного объекта и последствия ошибки для безопасности.

Алгоритм выбора платформы

Выбор аппаратной платформы представляет собой многокритериальную задачу. Для каждого кандидата измеряются задержка L, средняя мощность P, стоимость C, качество распознавания Q и запас памяти R. При одинаковых условиях тестирования можно определить нормированные оценки и интегральный критерий:

$$J = w_L \cdot L_norm + w_P \cdot P_norm + w_C \cdot C_norm + w_Q \cdot (1 - Q_norm), \quad \Sigma w_i = 1.$$

Минимизируемый критерий J удобен для предварительного ранжирования, но не должен заменять жёсткие ограничения: L_max ≤ T_budget, P ≤ P_limit, RAM_used ≤ RAM_available, Q ≥ Q_min. Весовые коэффициенты задаются требованиями конкретного проекта, а не выбираются произвольно после получения результатов.

Алгоритм выбора платформы Edge AI

Требования: задержка, мощность, цена

7 Практический пример: мобильный IoT-робот

Рассмотрим мобильного робота, который движется по технологическому коридору и распознаёт людей и препятствия перед собой. Камера передаёт видеопоток в вычислительный модуль; нейросеть выделяет объекты; алгоритм оценивает риск столкновения; микроконтроллер выполняет команды скорости и поворота. Для безопасного останова предусматриваются независимые датчики расстояния и аварийная блокировка.

Пусть в иллюстративном эксперименте измерены: захват 12 мс, предобработка 8 мс, инференс 28 мс, постобработка 6 мс, передача команды 4 мс и реакция привода 25 мс. Тогда $T_{total} = 12 + 8 + 28 + 6 + 4 + 25 = 83$ мс. При бюджете 100 мс номинальный запас составляет 17 мс. Эти значения являются учебными исходными данными, а не характеристиками конкретного устройства; для инженерного решения требуется измерить наихудшую задержку и её распределение.

При скорости $v = 0{,}6$ м/с робот за 83 мс проходит $s = v \cdot T = 0{,}0498$ м. Это расстояние добавляется к тормозному пути и запасу на погрешность обнаружения. Если задержка возрастёт до 180 мс, расстояние движения до реакции составит 0,108 м. Следовательно, производительность AI Hardware напрямую влияет на минимально допустимую дистанцию обнаружения.

7 Практический пример: мобильный IoT-робот (продолжение)

7.1. Архитектура экспериментального стенда

Камера → Jetson / альтернативная Edge AI-платформа → детектор объектов → блок оценки риска → интерфейс UART/CAN → микроконтроллер управления приводом → робот. Отдельные датчики расстояния → независимая логика безопасного останова. Телеметрия по MQTT → локальный сервер мониторинга. Применение IoT позволяет регистрировать метрики, но потеря связи с сервером не должна нарушать базовую безопасность.

8 Методика экспериментального сравнения

Для научно обоснованного сравнения выбирают одинаковую нейронную модель, одинаковые изображения, размер входа и целевую метрику качества. Затем на каждой платформе выполняют прогрев и серию повторных измерений. Отдельно фиксируют программные версии, температуру, частоту CPU/GPU, режим мощности, точность вычислений и способ подключения камеры. Если аппаратная платформа требует другой формат или квантование, необходимо явно указать различие и повторно измерить качество модели.

- **Время инференса:** медиана, 95-й и 99-й процентили, максимальное значение.
- Сквозная задержка от получения кадра до выдачи команды.
- Пропускная способность при одном и нескольких видеопотоках.
- Средняя и пиковая мощность, температура и устойчивость к троттлингу.
- Память и загрузка вычислительных блоков.
- Качество распознавания и число пропущенных опасных объектов.
- Воспроизводимость результатов не менее чем в нескольких независимых сериях.

8 Методика экспериментального сравнения (продолжение)

Результаты представляют в таблице с доверительными интервалами и графиками «задержка — мощность», «качество — задержка». Статистическую значимость различий оценивают с учётом дизайна эксперимента и распределения данных. При практическом внедрении добавляют тестирование при длительной непрерывной работе, вибрации, перепадах температуры и нарушениях связи.

9 Основные выводы

Современные интеллектуальные системы управления требуют согласованного выбора вычислительной архитектуры, модели машинного обучения и алгоритма управления. GPU и NPU ускоряют нейронные вычисления, но реальная эффективность определяется совместимостью программного стека и сквозными задержками. NVIDIA Jetson является одним из вариантов для сложных задач робототехники и компьютерного зрения; конкурирующие решения могут оказаться предпочтительнее по мощности, цене или специализации. Для промышленного внедрения обязательны проверка ограничений реального времени, устойчивости, воспроизводимости и независимых механизмов безопасности.

10 Контрольные вопросы

1. Какова роль AI Hardware в контуре интеллектуального управления?
2. Чем Edge AI отличается от облачного инференса и TinyML?
3. Какие вычислительные задачи наиболее эффективно выполняются на GPU?
4. В чём отличие NPU/TPU от универсального CPU?
5. Почему нельзя напрямую сравнивать TOPS разных устройств?
6. Какие компоненты входят в программную экосистему NVIDIA Jetson?
7. Для чего применяется TensorRT и какие риски связаны с квантованием?
8. Какие ограничения существуют при использовании Google Coral Edge TPU?
9. Как вычислить полную задержку интеллектуального контура?
10. Почему средняя задержка не гарантирует выполнение требований реального времени?
11. Какие факторы влияют на память нейронной модели?

10 Контрольные вопросы (продолжение)

12. Как оценивать энергопотребление на один вывод модели?
13. Какие метрики качества важны для детектора объектов на мобильном роботе?
14. Как организовать корректное экспериментальное сравнение Edge AI-платформ?
15. Почему функции аварийного останова нельзя полностью доверять нейросетевому распознаванию?

11 Рекомендуемая литература и документация

- **Ли П. Архитектура интернета вещей. М.:** ДМК Пресс, 2018.
- NVIDIA. Jetson Documentation и JetPack SDK Documentation (официальная документация). https://docs.nvidia.com/jetson/
- NVIDIA. TensorRT Documentation. https://docs.nvidia.com/deeplearning/tensorrt/
- Google Coral. Edge TPU Documentation. https://coral.ai/docs/
- Raspberry Pi. Documentation. https://www.raspberrypi.com/documentation/
- Intel. OpenVINO Documentation. https://docs.openvino.ai/
- OpenCV. Official Documentation. https://docs.opencv.org/

Примечание: численные технические характеристики устройств и совместимость программных версий необходимо сверять с актуальной документацией производителя для конкретной модификации.